



Intel 430MX PCIsset 82437MX (MTSC)/82438MX (MTDP) and 82371MX (MPIIX) Specification Update

Release Date: December 1997

Order Number 297656-004

The Intel 430MX PCIsset (MTSC), (MTDP) and (MPIIX) may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

Information in this document is provided in connection with Intel products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The Intel 430MX PCIset (MTSC), (MTDP) and (MPIIX) may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation

P.O. Box 5937
Denver CO 80217-9808

or call 1-800-548-4725
or visit Intel's website at <http://www.intel.com>

Copyright © Intel Corporation 1996, 1997.

- Third-party brands and names are the property of their respective owners.

CONTENTS

REVISION HISTORY	v
PREFACE	vi

Part I: Specification Update for Intel 82437MX (MTSC) and 82438MX (MTDP)

82437MX (MTSC) / (MTDP) GENERAL INFORMATION	9
82437MX (MTSC) / (MTDP) SPECIFICATION CHANGES	11
82437MX (MTSC) / (MTDP) ERRATA.....	12
82437MX (MTSC) / (MTDP) SPECIFICATION CLARIFICATIONS	21
82437MX (MTSC) / (MTDP) DOCUMENTATION CHANGES.....	22

PartII: Specification Update for Intel 82371MX (MPIIX)

82371MX (MPIIX) GENERAL INFORMATION	25
82371MX (MPIIX) SPECIFICATION CHANGES.....	27
82371MX (MPIIX) ERRATA	28
82371MX (MPIIX) SPECIFICATION CLARIFICATIONS.....	32
82371MX (MPIIX) DOCUMENTATION CHANGES	34

REVISION HISTORY

INTEL 430MX (MTSC), (MTDP) AND (MPIIX)

Date of Revision	Version	Stepping	Description
July 1996	-001		Initial Release
April 1997	-002		Added Errata 5 and 6
September 1997	-003		Added MTSC Errata 7
December 1997	-004		Added MTSC Documentation Change 3

PREFACE

This document is an update to the specifications contained in the Intel 430MX PCIset MTSC/MTDP Datasheet (290524), and the Intel 430MX PCIset MPIIX Datasheet (290525) and contains issues affecting all designs using the production revisions of the Intel 430MX PCIset.

It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools. It contains Specification Changes, Errata, Specification Clarifications, and Documentation Changes.

Nomenclature

Specification Changes are modifications to the current published specifications. These changes will be incorporated in the next release of the specifications.

Errata are design defects or errors. Errata may cause the Intel 430MX PCIset's behavior to deviate from published specifications. Hardware and software designed to be used with any given stepping must assume that all errata documented for that stepping are present on all devices.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Component Identification via Programming Interface

The 82437MX (MTSC) stepping can be identified by the following register contents:

82437MX Stepping	Vendor ID ¹	Device ID ²	Revision Number ³
A-1	8086h	1235h	01h
B-0	8086h	1235h	02h

The 82371MX (MPIIX) stepping can be identified by the following register contents:

82371MX Stepping	Vendor ID ¹	Device ID ²	Revision Number ³
A-1	8086h	1234h	01h
A-2	8086h	1234h	02h

NOTES:

1. The Vendor ID corresponds to bits 15-0 of the Vendor ID Register located at offset 00-01h in the PCI configuration.
2. The Device ID corresponds to bits 15-0 of the Device ID Register located at offset 02-03h in the PCI configuration space.
3. The Revision Number correspond to bits 7-0 of the Revision ID Register located at offset 08h in the PCI configuration space.

Part I:
Specification Update for
Intel 82437MX (MTSC)/82438MX (MTDP)

GENERAL INFORMATION

This section covers the **82437MX MTSC** and the **82438MX MTDP**.

82437MX MTSC

Stepping	S-Spec	Top Marking	Freq.	Notes
A-1	U036	SB82437MX S U036	60	Production
A-1	U037	SB82437MX S U037	66	Production
B-0	U069	SB82437MX66 S U069	66	Production

82438MX MTDP

Stepping	S-Spec	Top Marking	Freq.	Notes
A-1	—	FA82438MX	66	Production

SUMMARY TABLE OF CHANGES

The following table indicates the Specification Changes, Errata, Specification Clarifications, or Documentation Changes which apply to all currently available steppings and planned steppings. Intel intends to account for the outstanding issues through documentation or specification changes as noted. This table uses the following notations:

CODES USED IN SUMMARY TABLE

X:	Specification Change or Clarification that applies to this stepping or to this product line.
Doc:	Document change or update that will be implemented.
Fix:	This erratum is intended to be fixed in a future step of the component.
Fixed:	This erratum has been previously fixed.
NoFix	There are no plans to fix this erratum.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.
Shaded:	This erratum is either new or modified from the previous version of this document.

82437MX MTSC

NO.	A1	B0	PLANS	SPECIFICATION CHANGES
1	X	X	Doc	CAS[7:0]# Max. valid delay from HCLKIN rising changed to 9.0 ns (t44)
2	X	X	Doc	RAS[3:0]# pulse width low (CBR Refresh)
NO.	A1	B0	PLANS	ERRATA
1	X		Fix	During passive release the MTSC does not block CPU accesses to the write poster
2	X		Fix	Refresh period doubles leaving stop clock state or suspend
3	X		Fix	MA[0:1] buffers default to 2 mA
4	X	X	NoFix	MTSC does not insert a turnaround cycle under specific system scenarios
5	X	X	NoFix	Invalid memory cycles may occur entering Stop Clock or Stop Break.
6	X	X	NoFix	Short Trps exiting Stop Clock
7	X	X	NoFix	Invalid DRAM Cycles entering Stop Clock State
NO.	A1	B0	PLANS	SPECIFICATION CLARIFICATIONS
1	X	X	Doc	NA# test function does not disable pipelining
2	X	X	Doc	Layout considerations for MTSC/MTDP control signal interface
3	X	X	Doc	RAS# active timing during fast page mode accesses
NO.	A1	B0	PLANS	DOCUMENTATION CHANGES
1	X	X	Doc	MTSC package
2	X	X	Doc	PWROK, PWRSD, RTCCLK Vil and Vih
3	X	X	Doc	CACHE Control Register bits [5:4]



82437MX (MTSC) SPECIFICATION CHANGES

1. **CAS[7:0]# Max Valid Delay from HCLKIN Rising changed to 9.0 ns (t44)**

The CAS[7:0]# Max Valid delays from HCLKIN Rising have been changed as shown below:

Symbol	Parameter	66 MHz		60 MHz		Fig.	Notes
		Min	Max	Min	Max		
t44	CAS[7:0]# Valid Delay from HCLKIN Rising	1.5	9.0	1.5	9.0	7	0 pF

2. **RAS[3:0]# Pulse Width Low (CBR Refresh)**

The RAS[3:0]# Pulse Width Low (CBR Refresh) specifications have been changed as shown below:

Symbol	Parameter	66 MHz		60 MHz		Fig.	Notes
		Min	Max	Min	Max		
t40a	RAS[3:0]# Pulse Width Low (CBR Refresh)	5		5			HCLKs

82437MX (MTSC) ERRATA

1. *MTSC Does Not Block CPU Accesses to the Write Poster in Passive Release*

PROBLEM: When the MPIIX does a PHOLD# passive release to the MTSC arbiter, the MTSC is required to block any CPU-to-PCI cycles destined for the write buffers. Under certain passive release conditions these cycles are not blocked.

IMPLICATION: MPIIX will do a PHOLD# passive release to MTSC only if some PCI bus device retries the MPIIX DMA controller. If the MPIIX DMA controller does a memory transfer to the system memory behind MTSC, the MTSC will NEVER respond with retry. If the MPIIX DMA controller writes to system memory on the PCI bus that is NOT behind MTSC, there is a potential for the cycle to get retried.

WORKAROUND: This condition can be avoided if the DMA devices (either a DMA device on the Extended I/O bus or a docking ISA bus master) only transfers to system memory that is behind the MTSC DRAM controller.

STATUS: This erratum was fixed in the B-0 stepping of the MTSC.

2. *Refresh Period Doubles Leaving Stop Clock State or Suspend*

PROBLEM: When leaving the Stop Clock State or Suspend the Refresh state machine will ignore one edge of the refresh clock (the RTCCLK input on the MTSC). The refresh period doubles. (i.e. the refresh rate changes from 16 us to 32 us if originally programmed to 16 us.)

IMPLICATION: The DRAM refresh rate will be longer than the specified 16 us.

WORKAROUND: The system can be designed with a DRAM that accepts longer refresh cycles. Also the system can be modified to use a 64khz frequency to the RTCCLK input of the MTSC.

STATUS: This erratum was fixed in the B-0 stepping of the MTSC.

3. *MA[0:1] Buffers Default to 2 mA*

PROBLEM: A-1 silicon incorrectly drives MA[0:1] at 2 mA (should be 8 mA). The A-2 stepping has a new feature that increases the drive capability of the MA[2:11] buffers. When this feature is NOT enabled the MTSC should default to the A-0 configuration.

A-0 has 1 MA buffer drive setting: MA[0:1] at 8 mA and MA[2:11] at 2 mA

A-1 designed for 2 MA buffer drive settings: MA[0:1] at 8 mA and MA[2:11] at 2 mA (default to match A-0)
MA[0:1] at **8 mA** and MA[2:11] at 8mA (enabled by BIOS)

IMPLICATION: If the system is design with the MA[0:1] signals buffered from the MTSC then there is no issue. If not buffered, then the MA[0:1] signals might not have enough drive capability to meet DRAM timings.

WORKAROUND: Either buffer the MA[0:1] signals from MTSC or set all MA[0:11] signals to 8 mA (BIOS setting).

STATUS: This erratum was fixed in the B-0 stepping of the MTSC.



4. **MTSC Does Not Insert a Turnaround Cycle Under Specific System Scenarios**

PROBLEM: If a PCI master had been requesting the PCI bus for some time and the MTSC has just flushed its CPU-to-PCI write posting buffers, the MTSC could grant a master the bus one clock early. Under any other circumstance the MTSC grants the bus to the requesting master such that the MTSC floats the bus one clock before a master can start driving the bus. This is also a documented issue with 82437FX TSC and *no system failures or effects have been reported as a result of this issue.*

IMPLICATION: For the scenario described above, the MTSC can be in the process of floating its output buffers while a newly granted PCI master is starting to drive the bus. Therefore, some contention may be possible on the AD[31:0], C/BE[3:0]#, and PAR lines. Any possible contention is very limited due to the MTSC's max float delay timing of 8.5 ns.

WORKAROUND: A workaround is not recommended. For more detailed information, Intel has prepared a white paper titled "Analysis of MTSC Turnaround Issue." This document is available from your local Intel Sales Office.

STATUS: This erratum will not be fixed.

5. **Invalid memory cycles may occur entering Stop Clock state or Stop Break**

PROBLEM: Invalid DRAM cycles can occur when MTSC enters the Stop Clock state or when Stop Break occurs. Both Suspend refresh modes (CBR & Self Refresh) are affected by this issue. This condition affects only Stop Clock state and does not affect Stop Grant state. The Stop Clock state may still be used in systems which use 'self-refresh' for the suspend refresh mode, yet a boundary condition exists when entering the Stop Clock state in this mode that may cause a data error at physical address location 00 in the DRAM. Even though the probability of encountering this condition is small, it is recommended to save and restore location 00 upon entry/exit of Stop Clock state.

IMPLICATION: The invalid DRAM cycles can cause system malfunction.

WORK AROUND: CBR Refresh Mode

Do not use the Stop Clock State with CBR refresh mode. A workaround to allow use of the Stop Clock State with CBR suspend refresh mode is not available.

Note: Using only the Stop Grant state with CBR suspend refresh mode is acceptable.

5.2 **SELF-REFRESH MODE**

1. Use only self-refresh mode when Stop Clock State is implemented in the 430MX system. Set the Suspend Refresh Type bit to '1' in the MTSC DRAM Control Register (Offset 57h).
2. Save and restore the entire DRAM line at MA address 000h (8 bytes) to SMRam space as defined below:

Note: The 'Self-Refresh Save and Restore BIOS' is used in combination with the BIOS algorithm defined for the 'MPIIX CLKRUN# Does Not Function Correctly' Errata. See the 'Intel 430MX PCIsset Specification Update' available from your local Intel Sales Office.

5.2.1 **SELF-REFRESH SAVE AND RESTORE BIOS**

This BIOS work around **must** be used from within an SMI because physical location 0 access may not be allowed by current or future protected mode operating systems. The Stop Clock BIOS function should generate a SMI using the APMC write command and perform the Stop Clock entry from within the SMM with further SMIs disabled. The Stop Clock entry code for using the Stop Clock function is:

```

    pushf                ; save the interrupt state if needed
    cli                  ; Interrupts MUST be disabled before
                        ; entering Stopclock.

;    Protect the dword at physical address 0
;

    push    ds           ; save ds
    push    0            ; load 0 into ds
    pop     ds
    mov     eax, [0]      ; get dword to protect
    pop     ds           ; restore data seg
    mov     [BIOS_SAVE], eax ; save in sum

;
;    enter the stop clock mode
;

    mov     dx, ocf8h     ; point to config address
    mov     eax, 800008D4h ; bus=0, funct=0, dev=1, reg=D4h
    out     dx, eax       ; point to the CLK_CNTRL register
    mov     dx, 0cfch     ; point to config data
    in      al, dx        ; read CLK_CNTRL register
    mov     al, 09h       ; enable stop clock mode and clkrun#
    out     dx, al        ;
    in      al, 0b2h      ; APMC read. STPCLK# asserted here.
                        ; HCLK stops at this time. Wait for break
                        ; event.
                        ; restart and turn off CLKRUN# bit
    in      al, dx        ; dummy read to start PCI clock (count down starts
                        ; here.
    xor     al, al        ; set bits to clear stop clock mode bit and CLKRUN#
                        ; bit
    out     dx, al        ; actual write to disable CLKRUN (before 26 PCI
                        ; clks).

```



6. *Short Trps exiting stop clock*

PROBLEM: The RAS# Pre-charge spec. on self-refresh exit is violated while exiting Stop Clock.

This condition occurs if a refresh request (RTCCLK edge) is latched by the MTSC after the STPCLK# signal has been asserted and before the clocks are stopped. If the clocks stop before the MTSC has an opportunity to service the normal refresh request, the MTSC will switch the DRAM into self-refresh mode with a normal refresh pending.

When the system exits Stop Clock mode and takes the DRAM out of self refresh, the MTSC will immediately run the pending normal refresh only 2 HCLKs after deasserting the RAS# signal. The result is a RAS# in-active time that violates the T_{RPS} spec.

IMPLICATION: The result of this erratum causes read/write data error and sometimes system lockup for systems that support stop clock mode.

WORK AROUND: Systems which support stop clock mode require a software workaround which will prevent the refresh request from occurring during the stop clock entry sequence. This software workaround described is added to the stop clock routine in BIOS and prevents the fail condition. This software workaround programs the MTSC DRAMC register (offset 57h) bits [2:0] to a slower refresh rate. A delay of 22us should be added to force the refresh request to occur. This should prevent the next refresh from being latched by the MTSC before the clock stops.

The software for entering stop clock should look as follows:

```
;Code to store 22 usecs of CPU clock count.
start:
    mov     ax, seg del_100us
    mov     ds, ax
    mov     es, ax

; Determine cpu cycle to uS factor
get_reference:
    in      al,61h           ; read current contents
    and     al,0feh          ; turn off timer
    out     61h,al

;timer 2 uses 1.19 MHz oscillator
    mov     al,0b0h          ; timer 2 mode 0
    out     43h,al
    mov     al,077h          ; set for 100 us count
    out     42h,al
    mov     al,00
    out     42h,al
```

```

cli
db      0fh, 31h      ; save time stamp counter edx:eax into
                        ; ecx:ebx

mov     ebx,eax      ; save eax
mov     ecx,edx      ; save edx
in      al,61h       ; start the timer2
or      al,1
out     61h,al

wait_100us:
in      al,61h       ; read overflow
test    al,20h       ; see if overflow
jz      wait_100us
db      0fh,31h      ; save time stamp counter edx:eax
sub     edx,ecx      ; msdw just to set the carry, if needed
jne     get_reference
sub     eax,ebx      ; lsdw eax now has the number of cpu
                        ; clocks in 100 us

mov     dword ptr ds:[Del_100us],eax ; save the value for 100us
mov     edx,dword ptr ds:[Del_100us] ; get value to delay
mov     eax,20       ; number of uS to wait
mul     edx          ; factor in
mov     ecx,100      ; adjust to us
div     ecx
mov     dword ptr ds:[Del_22us], eax
sti

;Processor independent delay loop is accurate to .5%,resolution of 1 us.
;END of initialization

```




```
;code when entering STOP CLOCK
;change refresh rate here
    cli
    mov     eax,80000054h
    mov     dx,0cf8h
    out     dx,eax
    mov     al,13h          ; self refresh/125 usecs refresh
    mov     dx,0cffh
    out     dx,al          ; refresh modified here
    cli
    push    ecx
    push    ebx
    mov     ebx,dword ptr ds:[Del_22us]
    db      0fh,31h        ; read tsc
    add     ebx,eax
    mov     ecx,edx        ; match is now in cx:bx
    adc     ecx,0
wait1:
    db      0fh, 31h        ; read tsc
    cmp     edx,ecx        ; see if uppers match
    jb      wait1          ; not yet
    cmp     eax,ebx        ; see if lowers match
    jb      wait1
    pop     ebx
    pop     ecx
;end of 22 usecs delay
    mov     dx,0cf8h        ; point to config address
    mov     eax,800008D4h    ; bus=0, funct=0, dev=1, reg=D4h
    out     dx,eax          ; point to the CLK_CNTRL Register
    mov     dx,0cfch        ; point to donfig data
    in      al,dx           ; read CLK_CNTRL register
    mov     al,09h          ; enable stop clock mode and clkrun#
    out     dx,al          ;
```

```

        in      al,0b2h                ; APMC Read. STPCLK# asserted here.
                                         ; HCLK stops at this time. wait for break
                                         ; event. break event occurs here.
                                         ; MPIIX starts clocks. waits 1ms.
                                         ; deasserts STPCLK#.

;Resume from suspend here and change refresh rate back to normal
        nop
        nop
chg_refrshl:
        mov     eax, 80000054h
        mov     dx, 0cf8h
        out     dx, eax
        mov     al, 10h                ;self refresh/15.6 usecs refresh
        mov     dx, 0cffh
        out     dx, al                ;refresh modified here

;ORIGINAL CODE RESUMES HERE
        mov     dx,0cf8h                ; point to config address
        mov     eax,800008D4h           ; bus=0, funct=0, dev=1, reg=D4h
        out     dx,eax                 ; point to the CLK_CNTRL Register
        mov     dx,0cfch                ; point to config data
        in      al,dx                  ; dummy read to start PCI clock
                                         ; (count-down starts here)
        xor     al,al                  ; set bits to clear stop clock mode bit
                                         ; and CLKRUN# bit
        out     dx,al                  ; actual write to disable CLKRUN

```

STATUS: See the Summary Table of Changes.



7. *Invalid DRAM cycles entering Stop Clock State*

PROBLEM: If a break event occurs after the self refresh entry sequence has started as a result of a Stop Clock mode entry, but before the self refresh entry sequence has completed, the MTSC DRAM Controller may generate DRAM cycles which violate the DRAM specification at the start of the next stop clock mode entry.

IMPLICATION: For systems using the Stop Clock mode, if this condition occurs, unexpected behavior will result. The most likely failure mechanism for this erratum is a system hang, but it is possible that data corruption may result. Intel has observed this erratum only in a test environment under a forced alignment of events.

WORKAROUND: It is necessary to put the DRAM controller into self refresh mode using the suspend refresh mechanism prior to entry to stop clock mode. BIOS should ensure that break events are enabled then execute the following sequence. Note: the Short Trps workaround is not needed if this method is used. (errata 6)

1. Set SUSREF=1. (Suspend must not be enabled in MPIIX during stop clock routine)
2. Delay Loop until MTSC completes self refresh entry sequence (~32us).
3. Enter Stop Clock (Set Stop Clock Mode, enable CLKRUN#, APMC read) in the MPIIX
...systems remains in stop clock state until break event...then
MPIIX starts clocks and deasserts CLKRUN# (MTSC still in self refresh via the SUSREF=0.)
MPIIX waiting 1ms then deasserts STPCLK#.
4. Disable CLKRUN#
5. Set SUSREF=0 to exit MTSC from self refresh mode and start normal CBR.

The critical code needed after setting susref until clearing susref must be dword aligned and less than 16 bytes to ensure it fits in the L1 cache. The prefetch queue must be flushed before executing beyond this 16 byte region by performing a cpu id instruction. The clock run bit must be cleared within 26 pci clocks or 52 hclocks after the first PCI cycle occurs. Pseudo code follows.

```
stop_clock()
{
    clear_interrupts();           // interrupts must be cleared
    temp = read_mpiix ( 0xD4 ) & 0x80; // get the clock control register
    write_mpiix ( 0xD4, temp | 0x9); // set the stpcl mode, enable clock run
    temp = read_mpiix ( 0xCC ) & 0xf0; // preserve the upper bits
    asm      align      16           // must be dword aligned to ensure entire cache line
                                           // start of 16 byte restriction

    write_mpiix ( 0xCC, temp | 8); // set the susref bit
    delay_us(32);                  // wait 32 us to let the self refresh occur
    read_io (0xB2);                // do stop clock until break event

    write_mpiix ( 0xCC, temp ); // clear the susref bit
    // end of 16 byte restriction, memory is now fully functional
    // start of 26 pci clocks to clearing the clock run
    temp = read_mpiix ( 0xD4 ) & 0x80; // get the clock control register
    write_mpiix ( 0xD4, temp ); // disable clock run
    // end of 26 PCI clock restriction
    enable_interrupts();
}
```

STATUS: This erratum will not be fixed.



82437MX (MTSC) SPECIFICATION CLARIFICATIONS

1. ***NA# Test Function Does Not Disable Pipelining***

The MTSC A0 step implements a reserved function to provide the ability to disable pipelining for factory test purposes. The NA# disable test bit (*MTSC offset 52h, bit 3*) does not disable pipelining when set.

The MTSC A-0 Step NA# signal must always be connected to the processor.

Because this function is intended for factory testing of the MTSC, it is not considered a product erratum.

2. ***Layout Considerations for MTSC/MTDP Control Signal Interface***

The MTSC to MTDP interface signals, MOE#, HOE#, POE#, and MADV# should be routed from the MTSC to the MTDP such that the MTDPs are symmetrically "T"ed off. Damping resistors (22 ohm) should be placed on these lines close to the MTSC to ensure signal quality.

3. ***RAS# Active Timing during Fast Page Mode Accesses***

DRAM specifications include maximum RAS# active time during fast page mode accesses. The MTSC has no direct means to ensure this specification is not violated. The DRAM Extended Refresh Rate (DRR) bits located in the DRAM Control (DRAMC) register (offset 57h) should be programmed to ensure that 4 refresh requests occur within this DRAM maximum RAS# active time specification. The fast page mode accesses will be broken when the fourth refresh request is queued. This will ensure the deassertion of RAS# within the required DRAM specification. The conditions required to exceed the RAS# active time typically only occur in special software used to conduct memory performance testing. See section 4.3.4 of MTSC EDS for more detail on these conditions.

For example:

DRAM Max. RAS# Pulse Width (fast page mode) = 100 us

DRAMC DRR bits (2:0) programmed to 000 = 15.6 us Refresh Rate

This ensures that 4 refresh requests will be generated and queued, page mode broken, and RAS# lines deasserted within the 100 us Max RAS# Pulse Width specification of the DRAM.

82437MX (MTSC) DOCUMENTATION CHANGES

1. *MTSC Package Typos*

1. The "e1 (lead pitch)" for 100TQFP should be 0.5 nom, not 0.05 nom.
2. The MTSC package name should be '208 lead SQFP.'

2. *PWROK, PWRSD, RTCCLK Vil and Vih*

Add the following to Table 1. MTSC D.C Characteristics:

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL2}	Input Low Voltage	0.0	0.2 VDDR	V	Note 7
V_{IH2}	Input High Voltage	0.8 VDDR	VDDR + 0.3	V	Note 7

Note 7: V_{IL2} and V_{IH2} apply to the following signals: PWROK, PWRSD, RTCCLK. The voltage reference is to the VDDR "resume well" pin voltage.

3. *Cache Control Register Bits [5:4]*

The "Intel 430MX PCIsset 82430MX Mobile System Controller (MTSC) and 82438MX Mobile Data Path (MTDP)" databook, Order Number 290534-001, dated April 1996, is changed as follows:

On page 22, Section 3.2.10, CC - Cache Control Register, bits [5:4], the SRAM Type (SRAMT) description shows "Bits [7:6]". This is incorrect. The table should say "Bits [5:4]".



Part II:

Specification Update for 82371MX (MPIIX)

GENERAL INFORMATION

This section covers the 82371MX (MPIIX).

Component Markings

82371MX MPIIX

Stepping	S-Spec	Top Marking	Freq.	Notes
A-1	U034	FA82371MX S U034	60	Production
A-1	U035	FA82371MX S U035	66	Production
A-2	U067	FX82371MX66 S U067	66	Production

SUMMARY TABLE OF CHANGES

The following table indicates the Specification Changes, Errata, Specification Clarifications, or Documentation Changes which apply to all currently available steppings and planned steppings. Intel intends to account for the outstanding issues through documentation or specification changes as noted. This table uses the following notations:

CODES USED IN SUMMARY TABLE

X:	Specification Change or Clarification that applies to this stepping or to this product line.
Doc:	Document change or update that will be implemented.
Fix:	This erratum is intended to be fixed in a future step of the component.
Fixed:	This erratum has been previously fixed.
NoFix	There are no plans to fix this erratum.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.
Shaded:	This erratum is either new or modified from the previous version of this document.

82371MX MPIIX

NO.	A1	A2	PLANS	SPECIFICATION CHANGES
1	X	X	Doc	IDE PIO mode 2 interface timings not supported
2		X	Doc	IDE timings: DD[15:0] setup and hold timings
NO.	A1	A2	PLANS	ERRATA
1	X		Fix	Group enable for local traps does not properly enable IDE trap
2	X		Fix	MPIIX PC/PCI REQ#/GNT# may not be arbitrated fairly
3	X	X	NoFix	MPIIX CLKRUN# does not function correctly
4	X		Fix	ACT mode. Burst clock timer is not reloaded
5	X		Fix	DMA restrictions for PC/PCI and F-type
6	X	X	NoFix	Enabled system events cause stop break events
7	X	X	NoFix	IRQ8# system event and stop break event can not be disabled
8	X		Fix	EXTEVNT# has reversed polarity (active high)
9	X	X	NoFix	IRQ12/M cannot be enabled as system SMI event
10	X	X	NoFix	Active but masked IRQ causes system event, prevents STPCLK# assertion
11	X	X	NoFix	SM_EN bit does not disable local trap function
NO.	A1	A2	PLANS	SPECIFICATION CLARIFICATIONS
1	X	X	Doc	Power plane control, pin states in suspend
NO.	A1	A2	PLANS	DOCUMENTATION CHANGES
1	X	X	Doc	Audio and FM synthesis I/O access trap ranges
2	X	X	Doc	MPIIX package
3	X	X	Doc	MPIIX electrical specifications: RTCCS, PIRQ
4	X	X	Doc	IRQ8# buffer is 5/3V CMOS Schmitt, not 5V TTL Schmitt
5	X	X	Doc	PWROK is 5/3V CMOS Schmitt, not 5V CMOS Schmitt
6	X	X	Doc	Additional Vil and Vih specifications

82371MX (MPIIX) SPECIFICATION CHANGES

1. IDE PIO Mode 2 Interface Timings Not Supported

PIO Mode 2 IDE interface timings are no longer supported by MPIIX.

In one test case, it was found that an older PIO Mode 2 IDE drive failed to operate under the MPIIX Mode 2 interface timings. Additional Mode 2 IDE drives were tested which did not fail.

Currently, drive vendors supply IDE drives which support up to PIO Mode 4 timings and no longer supply PIO Mode 2 only drives. Because of this, Mode 2 IDE timings are considered to be obsolete and it is felt that Mode 2 timings support is not necessary.

2. IDE Timings Change for DD[15:0] Setup and Hold

Replace the values in Table 6 as shown below with those in Table 6a.

Table 6.

Symbol	Parameter	33 MHz		Units	Notes	Fig.
		Min	Max			
t103	DD[15:0] Setup to PCICLK	10		ns		24
t104	DD[15:0] Hold from PCICLK	2		ns		24
t110	DIOx# Valid Delay from PCICLK Rising	2	20	ns		24

Table 6a.

Symbol	Parameter	33 MHz		Units	Notes	Fig.
		Min	Max			
t103a	DD[15:0] Setup to DIOR#	6		ns		24
t104a	DD[15:0] Hold from DIOR#	5		ns		24
t110	DIOx# Valid Delay from PCICLK Rising	2	10	ns		24

82371MX (MPIIX) ERRATA

1. *Group Enable for Local Traps Does Not Properly Enable IDE Trap*

PROBLEM: When the IDE local trap is enabled (Reg 0ab_h), but the group bit is disabled (Reg 0C3_h), an I/O cycle to the IDE controller is not claimed by MPIIX. This does not happen with any other device, only the IDE.

IMPLICATION: When both the group enable and the individual enable for the IDE are '1', any IDE cycle is target aborted and an SMI is generated. When the group bit is clear and the local IDE bit is set, the cycle is terminated with a master abort.

WORKAROUND: In order to disable all traps, disable the individual IDE enable bit as well as the group enable bit.

STATUS: This erratum was fixed in the A-2 stepping of the MPIIX.

2. *MPIIX PC/PCI REQ#/GNT# May Not Be Arbitrated Fairly*

PROBLEM: The PC/PCI REQ# (REQA# or REQB#) deasserting and reasserting within 2 clks causes the MPIIX arbiter to lock onto that REQx# and not rearbitrate. That REQx# is granted the bus twice consecutively when it is configured in PCI DMA non-expansion.

IMPLICATION: Impact is low since there are currently no devices that will use a dedicated DMA channel for the PCI DMA. Card bus implementations will use this PC/PCI REQ#/GNT# pair in expansion mode.

WORKAROUND: No workaround is available.

STATUS: This erratum was fixed in the A-2 stepping of the MPIIX.

3. *MPIIX CLKRUN# Does Not Function Correctly*

PROBLEM: The CLKRUN# feature does not re-start properly.

IMPLICATION: The CLKRUN# must be disabled for normal operation. It must only be enabled for MPIIX to put the system in a CPU Stop Clock State.

WORKAROUND: The CLKRUN# feature should be enabled in the normal fashion to enable the stop clock state. Upon wake-up the CLKRUN# feature should be disabled before the PCI clock is allowed to automatically stop (in about 26 PCI clocks). The following code has been verified to bring the system out of the stop clock state correctly.

```

mov     dx,0cf8h           ; point to config address
mov     eax,800008D4h      ; bus=0, funct=0, dev=1, reg=D4h
out     dx,eax             ; point to the CLK_CNTRL Register
mov     dx,0cfch           ; point to donfig data
in      al,dx              ; read CLK_CNTRL register
mov     al,09h             ; enable stop clock mode and clkrun#
out     dx,al              ;
in      al,0b2h            ; APMC Read. STPCLK# asserted here. HCLK stops at this
                           ; time. wait for break event. break event occurs here.
                           ; MPIIX starts clocks. waits 1ms. deasserts STPCLK#.
mov     dx,0cfch           ; point to config data
in      al,dx              ; dummy read to start PCI clock (count-down starts here)
xor     al,al              ; set bits to clear stop clock mode bit and CLKRUN# bit
out     dx,al              ; actual write to disable CLKRUN (before 26 PCI clks.)

```

STATUS: This erratum will not be fixed.

4. *ACT Mode Burst Clock Timer is Not Re-Loaded Correctly.*

PROBLEM: In Auto Clock Throttle mode the Burst Clock Events will not correctly re-load the Burst Clock Timer (BSTCLK_TMR).

IMPLICATION: The ACT break events that forced a short burst of CPU activity (BSTCLK Events reload the BSTCLK_TMR) must be enabled to use the long burst timer (CLKTHL Events reload the CLKTHL_TMR). This means that the CPU will wake up for a longer burst of activity (8 ms to 8 sec) than is necessary for the events that only require a short burst of activity (32 us to 8 mS).

WORKAROUND: Enable all ACT break events as long burst events (CLOCK THRTL Break Events).

STATUS: This erratum was fixed in the A-2 stepping of the MPIIX.

5. *PC/PCI DMA Does Not Function Correctly in Demand Mode.*

PROBLEM: MPIIX increments the memory address too early during PC/PCI DMA Demand Mode transfers.

IMPLICATION: PC/PCI DMA cannot be used for Demand Mode transfers from the ISA cards in the docking station and PCI to PCMCIA adapters that support the PCMCIA Rev 2.1 DMA specification.

WORKAROUND: No workaround is available.

STATUS: This erratum was fixed in the A-2 stepping of the MPIIX.

6. *Enabled System Events Cause Stop Break Events*

PROBLEM: In Stop Clock mode when the HCLK & PCICLK are stopped (through APMC), an occurrence of a system event (enabled as a system event but NOT as a break event) breaks the system out of Stop Clock mode.

IMPLICATION: Events which are not desired to break the system out of Stop Clock mode will do so, lowering the power management efficiency. This has no effect on Stop Grant functionality.

WORKAROUND: The SMI handler must disable events as system events which are not enabled as break events before entering the stop clock mode (through APMC). The SMI handler needs to re-enable these events as system events after coming out of stop clock mode.

STATUS: This erratum will not be fixed.

7. *IRQ8# Stop Break Event Cannot be Disabled.*

PROBLEM: In Stop Clock or Stop Grant mode, an assertion of IRQ8# breaks the system out of Stop Clock or Stop Grant mode.

IMPLICATION: IRQ8# assertion will cause system to break out of Stop Clock or Stop Grant states when not desired to do so. IRQ8# is typically connected to the Real Time Clock controller for use as an alarm function. It is generally used to power up a shutdown system to perform a specific function at a desired time. In this case, the Stop Clock or Stop Grant break is desired.

WORKAROUND: If the system software desires to not break from Stop Clock or Stop Grant state, it should disable IRQ8# generation at the device connected to IRQ8# (typically the RTC controller).

STATUS: This erratum will not be fixed.

8. *EXTEVNT# Has Reversed Polarity (Active High)*

PROBLEM: EXTEVNT# has reversed polarity internal to MPIIX.

IMPLICATION: Improper system operation if EXTEVNT# is enabled.

WORKAROUND: System designer should add external inverter to signal connected to EXTEVNT# pin if EXTEVNT# events are desired. If EXTEVNT# events are not going to be used, software should ensure that they are disabled in BSTCLK_EVTN_EN_X and CLKTHL_BRKEVNT_EN_X (X = 0:6) registers. In this case no external inverter is required. The EXTEVNT# signal is level triggered.

STATUS: This erratum was fixed in the A-2 stepping of the MPIIX.

9. *IRQ12/M Cannot Be Enabled As System SMI Event*

PROBLEM: IRQ12/M signal cannot be enabled as a System SMI event in SYS_SMI_ENABLE register.

IMPLICATION: In the state of Video controller and display being in Local Standby, a movement of the mouse will generate an IRQ12/M. This would normally cause an SMI to occur resulting in video being restored.. Another method is required to ensure generation of SMI to allow video restoration.

WORKAROUND: The SMI handler should enable a Local Trap to the Keyboard Controller address range and ensure that IRQ12/M interrupt is not masked. The IRQ12/M signal will then cause an interrupt. When program control goes to the interrupt handler, it will access the keyboard controller to check the status of the interrupt. This access will cause a local trap to occur. This local trap will in turn generate an SMI, which will result in restoration of video.

STATUS: This erratum will not be fixed.

10. *Active but Masked IRQ Causes System Event and Stop Break Event*

PROBLEM:

1. If enabled as a System Event, an active but masked IRQ will cause a System Event resulting in continuous timer reload of the Global Standby Timer, the Software/External SMI# Delay Timer, and the Suspend SMI# Delay Timer. This prevents SMI requests from being generated from any of these sources.
2. If enabled as a Stop Break event, a Burst Clock event (ACT), or a Clock Throttle event (ACT), an active but masked interrupt will also generate a continuous event, preventing CLKRUN# or STPCLK# assertion.

IMPLICATION: In the case of a driver which polls IO devices, an interrupt may be generated by the IO device with the IRQ masked internal to MPIIX.

1. The interrupt is never serviced but causes a continual reload of the above mentioned delay timers preventing their generation of an SMI request.
2. This interrupt also prevents stopping of PCICLK or HCLK.

WORKAROUND: Please contact your local Intel Sales Office for the latest documentation regarding this erratum. Ask for the "MPIIX Active but Masked IRQ" workaround document.

STATUS: This erratum will not be fixed.

11. *SM_EN Bit Does Not Disable Local Trap Function*

PROBLEM: The SM_EN bit (bit 1 of SYS_MNGT_CNTRL register, PCI offset C0h) does not disable the Local Trap function.

IMPLICATION: If SMI generation is disabled with Local Traps enabled, an IO cycle to one of the trap enabled address ranges will not complete to its destination, will be terminated on the PCI bus with a target abort, and will be returned to the CPU with BRDY#. The SMI# will not be generated to the CPU indicating that an IO restart is necessary and the IO cycle will never be properly completed.

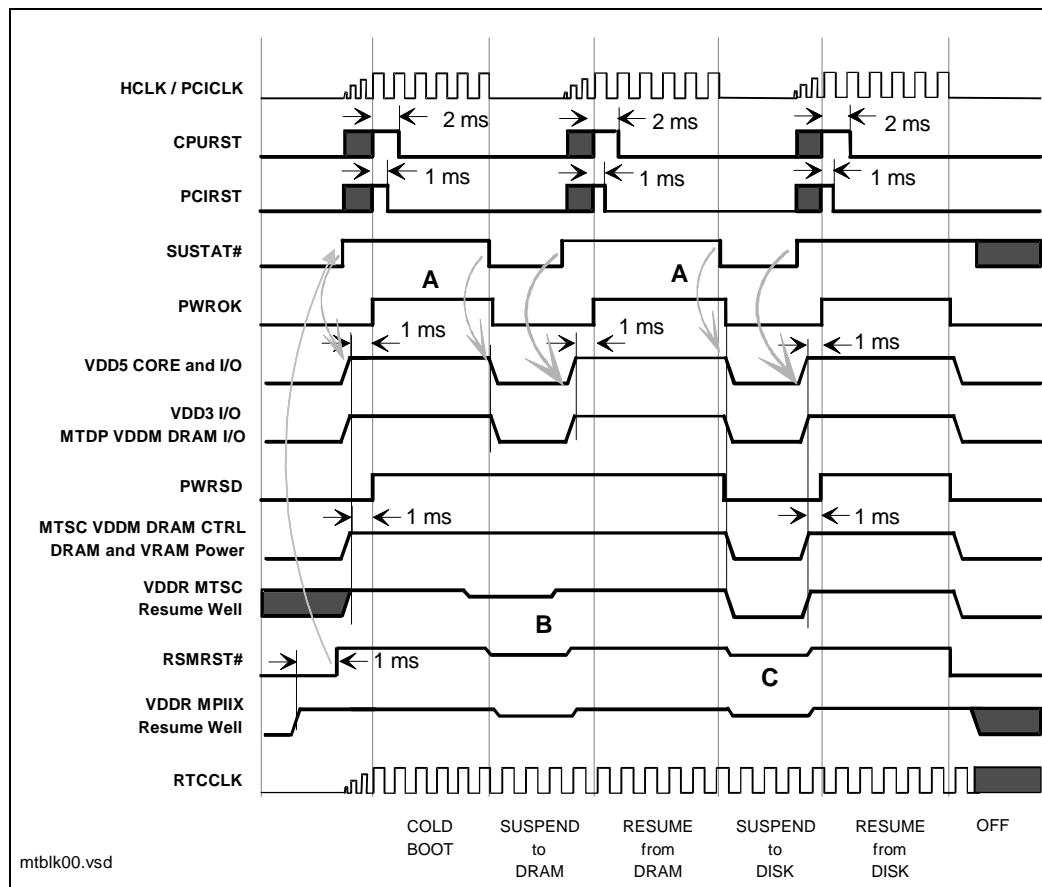
WORKAROUND: When the SM_EN bit is disabled (set to 0), the LTRP_SMI_EN bit (bit 4 of GLOBAL_SMI_ENABLE register, PCI offset C3h), must also be disabled (set to 0).

STATUS: This erratum will not be fixed.

82371MX (MPIIX) SPECIFICATION CLARIFICATIONS

1. Power Plane Control

The following diagram provides a valid SUSTAT# signal at cold boot in order to turn on the power plane:



The following table shows the state of the Intel 430MX PCIsset signals during the different system states.

MTSC	Normal	Suspend-DRAM	Suspend-DISK
MA[11:0]	Normal	Driven-LOW	Not-Powered
WE#	Normal	Driven-HIGH	Not-Powered
RAS#	Normal	Active	Not-Powered
CAS#	Normal	Active	Not-Powered
PWRSD	5V	3.3V	0V
VDDR	5V	5V or 3.3V	0V
VDDM	3.3V	3.3V	0V
VDD5	5V	0V	0V
VDD3	3.3V	0V	0V
Other	Normal	Not-Powered	Not-Powered

MTDP	Normal	Suspend-DRAM	Suspend-DISK
VDD5	5V	0V	0V
VDD3	3.3V	0V	0V
VDDM	5V or 3.3V	0V	0V
Other	Normal	Not-Powered	Not-Powered

MPIIX	Normal	Suspend-DRAM	Suspend-DISK
IRQ8	Normal	Normal	Normal
COMRI	Normal	Normal	Normal
SRBTN#	Normal	Normal	Normal
EXTSMI#	Normal	Normal	Normal
BATLOW#	Normal	Normal	Normal
RTCCLK	Active	Active	Active
RTCCLKO	Active	Active	LOW
RSMRST#	VDDR	VDDR	VDDR
SUSTAT#	HIGH	LOW	LOW
5V	5V OR 3.3V	5V or 3.3V or RTCVCC	
5V	0V	0V	
3.3V	0V	0V	
Others	Normal	Not-Powered	Not-Powered

82371MX (MPIIX) DOCUMENTATION CHANGES

1. **Audio and FM Synthesis I/O Access Trap Ranges**

Accesses to I/O ranges 2xAh and 2xEh trap I/O reads *and* writes when enabled.

2. **MPIIX Package Typos**

1. The top view of the 176 TQFP is showing the diagram for 100 TQFP. That is, each side should show 44 pins instead of 25 pins.
2. The "e1 (lead pitch)" for 100TQFP should be 0.5 nom, not 0.05 nom.

3. **MPIIX Electrical Specifications: RTCCS, PIRQ**

1. The Chip Select timings t68a and t68b (page 11, table 4) also apply to the signal RTCCS#.
2. The PCI Interface timings t91 and t92 apply to signal PIRQ[A,B], not PIRQ[3:0].

4. **RQ8# Buffer is 5/3V CMOS Schmitt not 5V TTL Schmitt**

1. The IRQ8# Buffer type should read 5/3V CMOS Schmitt vice 5V TTL Schmitt.

5. **PWROK is 5/3V CMOS Schmitt not 5V CMOS Schmitt**

1. The PWROK Buffer type should read 5/3V CMOS Schmitt vice 5V CMOS Schmitt.

6. **Additional Vil and Vih Specifications**

Add the following to Table 1. MPIIX D.C Characteristics:

Symbol	Parameter	Min	Max	Unit	Notes
V _{IL2}	Input Low Voltage	0.0	0.2 V _{DDR}	V	Note 7
V _{IH2}	Input High Voltage	0.8 V _{DDR}	V _{DDR} + 0.3	V	Note 7

Note 7: V_{IL2} and V_{IH2} apply to the following signals: PWROK, RTCCLK, RTCCLKO, SUSTAT#, RSMRST#, BATLOW#, SRBTN#, COMRI#, EXTSMI#, IRQ8#. The voltage reference is to the V_{DDR} "resume well" pin voltage.